CS 341, Winter 2021 A. Storjohan, H. Zhou

PROGRAMMING ASSIGNMENT 1

DUE: Wednesday, March 3, 11:59 PM. DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

Please read `http://www.student.cs.uwaterloo.ca/~cs341` for general instructions and policies.

1. [20 marks] In this programming assignment you will implement your backtracking algorithm for the Treasure Collector Problem from Assignment 4, Question 2. Here is a reminder of the question:

   Treasure Collector. Suppose your co-op company asks you to design a video game called *Treasure Collector*, where the mission of a player is to collect treasures from a castle with h floors. The game starts on the 1st floor, and there are n rounds in total. In the i-th round, a treasure with value $v_i \geq 0$ will appear on one of the h floors, and it will disappear after this round. The player can only collect the treasure on her current floor. Before each round starts, the player can decide whether she wants to go up to some higher floor or stay in the same floor. Notice that the player can never go back to a lower floor. An additional constraint is that the player's bag can only contain at most $m \leq n$ items, so the goal of the player is to collect at most m treasures to maximize the total value without violating the collecting rules.

   As the game designer, you want to display the best possible total value a player can collect at the end of the game, so that the players know how far they are away from the optimal value. Formally, you are given the number of rounds n, the bag limit $m \leq n$, the height of the castle h, and an array $[(v_1, f_1), \dots, (v_n, f_n)]$ as input, where $v_i \geq 0$ is the value of the i-th treasure which is located on the f_i-th floor for $f_i \in \{1, \dots, h\}$. Notice that $1 \leq i_1 < i_2 < \dots < i_j \leq n$ is a set of valid collection if and only if $f_{i_1} \leq f_{i_2} \leq \dots \leq f_{i_j}$ and $j \leq m$. Design a dynamic programming algorithm to find a valid collection with the maximum total value that a player can collect. The running time of your algorithm should be polynomial in n, m and h.

   Write your code in C++ or Python. Your program will be tested to see if your output produces the maximum total value that a player can collect and satisfies the constraints on the collection set $1 \leq i_1 < i_2 < \dots < i_j \leq n$ (note that the set need not be unique, so we will not test for an exact match).

   Input and Output. Your program should read from standard input and write to standard output.

   The input consists of $n + 1$ lines. The first line will contain 3 whitespace delimited positive integers which are n, m and h. Then, n lines follow where the ith line contains 2 whitespace delimited positive integers that are v_i and f_i.

   The output contains 2 lines. The first should be just an integer representing the maximum total value that a player can collect. The second should contain a list of whitespace delimited numbers indicating the collection set $\{i_1, i_2, \dots, i_j\}$.

Examples.

(a) For the following input:

```
5 2 10
8 3
6 4
1 1
6 5
9 2
```

the output is:

```
14
1 4
```

(b) For the following input:

```
3 2 100
2 32
8 13
9 83
```

the output is:

```
17
2 3
```

Submission Guidelines.

- Programming assignments are submitted through Marmoset: https://marmoset.student.cs.uwaterloo.ca/
- Your program will be compiled / run in the student Linux environment using the command "g++ -std=c++14" for C++ and "python3" for Python.
- Submit one zip file containing only your code file which must be named "prog1.cpp" or "prog1.py".
- Your score will be the score of your best submission, which depends on a set of secret testcases.

  You will be able to see two small testcases as a sanity check for your code. In addition, there will be a third public testcase which will tell you if your code times out in the largest testcase (but it doesn't test correctness).
- Finally, you can expect the input to satisfy that $n, m, h, v_i, f_i \leq 10^3$.